



# Boosting Kernel Combination for multi-class image categorization

Alexis Lechervy, Philippe-Henri Gosselin, Frédéric Precioso

## ► To cite this version:

Alexis Lechervy, Philippe-Henri Gosselin, Frédéric Precioso. Boosting Kernel Combination for multi-class image categorization. 2012 IEEE International Conference on Image Processing (ICIP), Sep 2012, Orlando, United States. pp.4. hal-00753156

**HAL Id: hal-00753156**

**<https://hal.science/hal-00753156>**

Submitted on 17 Nov 2012

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# BOOSTING KERNEL COMBINATION FOR MULTI-CLASS IMAGE CATEGORIZATION

*Alexis Lechervy, Philippe-Henri Gosselin*

ETIS, CNRS, ENSEA, Univ Cergy-Pontoise  
6 avenue du Ponceau, BP44  
F95014 Cergy-Pontoise, France  
{alexis.lechervy,gosselin}@ensea.fr

*Frédéric Precioso*

I3S - UMR7271 - UNS CNRS  
2000, route des Lucioles,  
F06903 Sophia Antipolis - France  
precioso@polytech.unice.fr

## ABSTRACT

In this paper, we propose a novel algorithm to design multi-class kernel functions based on an iterative combination of weak kernels in a scheme inspired from boosting framework. The method proposed in this article aims at building a new feature where the centroid for each class are optimally located. We evaluate our method for image categorization by considering a state-of-the-art image database and by comparing our results with reference methods. We show that on the Oxford Flower databases our approach achieves better results than previous state-of-the-art methods.

**Index Terms**— Image databases, Machine learning algorithms, Boosting

## 1. INTRODUCTION

Recent machine learning techniques have demonstrated their power for classifying data in challenging contexts (large databases, very small training sets or on the opposite, huge training sets, dealing with user interactions...). However, emerging problems are pushing these methods away their limits with several hundred of image categories to be classified, with millions of images both in training and testing datasets.

Different frameworks have been proposed to address image categorization task. One can consider kernel frameworks based on SVM [1] or based on boosting [2]. In this article, we propose a new method combining these two frameworks: we design a scheme inspired from boosting framework to build kernel functions which are then integrated in a SVM.

Different techniques have been proposed, in the literature, to linearly combine several (minor) kernels leading to improved performances of the hence designed major kernel. Among these methods, the last decade has seen an important increase of interest of methods which try to learn the combination of kernels in a so-called Multiple Kernel Learning framework (MKL) [3]. In the classic MKL framework, methods aim at jointly optimizing the weight of each kernel

in the combination. However, the two currently most powerful MKL methods are not following this exact scheme: Gehler and Nowozin [4] proposed a boosting-oriented scheme optimizing alternately the combination weights and the combined kernels which reached the best performance on classic MKL evaluation databases. But more recently, Orabona et al. [5] achieve the best current performances by proposing to change the MKL problem formulation.

In the proposed method, we are inline with the works done by Crammer et al. [6], Kawanabe et al. [7] and Gehler et al. [4]. We propose to design a linear combination of base kernels using the boosting paradigm, similarly to [6]. However, we focus on a strategy for multi-class learning using many different features. We also take inspiration from [7] taking advantage of kernel centered alignment to compute the combination weights but they compute all the combination weights at once while we do it iteratively. We consider at each iteration their combination weight computation approach in order to determine the weight in the combination of the major kernel at previous iteration and the weak kernel to be added.

In section 2, we introduce the context of linearly combine kernels inside a boosting framework. In section 3 we develop in details our method. Section 4 presents experiments on image categorization in referenced database for the MKL context and show that our method outperforms the current state-of-the-art approaches. Then we conclude and present of some perspectives of this work.

## 2. LINEAR KERNEL COMBINATION USING BOOSTING

### 2.1. Linear combination

The aim of this paper is to design a kernel function  $K(.,.)$  as a linear combination of base kernel functions  $k_t(.,.)$ :

$$K_T(\mathbf{x}_i, \mathbf{x}_j) = \sum_{t=1}^T \beta_t k_t(\mathbf{x}_i, \mathbf{x}_j),$$

where  $\mathbf{x}_i$  is the features for image  $i$ .

This combination is performed to match a target kernel function  $K^*$ , whose values  $\mathbf{K}^*$  are sampled on a given training set. In the following, we mix functions  $(f, K, \dots)$  and their values on the training set (written in bold  $\mathbf{f}, \mathbf{K}, \dots$ ). The combination is evaluated thanks to Kernel Target Alignment, a popular criterion for kernel optimization. More specifically, we use the centered alignment which allows kernel matrices with unbalanced classes [7]:

$$\mathcal{A}_{\mathbf{H}}(\mathbf{K}_T, \mathbf{K}^*) = \frac{\langle \mathbf{H}\mathbf{K}_T\mathbf{H}, \mathbf{H}\mathbf{K}^*\mathbf{H} \rangle}{\|\mathbf{H}\mathbf{K}_T\mathbf{H}\| \|\mathbf{H}\mathbf{K}^*\mathbf{H}\|},$$

with  $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$  and  $\mathbf{1}$  the vector full of ones.

## 2.2. Target kernel

The target kernel matrix  $\mathbf{K}^*$  is defined from the  $n \times n_{c'}$  matrix  $\mathbf{L}$  of labels on the training set (1 for positive image and -1 otherwise):

$$L_{i,c} = \text{label of image } i \text{ for category } c.$$

To overcome the dependence between classes, we propose to consider the matrix  $\mathbf{Q}$  of the  $QR$  decomposition of  $\mathbf{H}\mathbf{L}$ . We select only the columns where the diagonal element of  $\mathbf{R}$  is not null. Thus  $\mathbf{Q}$  is a  $n \times n_c$  full rank matrix, assuming that all classes are independent. Our target Gram matrix is then defined as  $\mathbf{K}^* = \mathbf{Q}\mathbf{Q}^\top$ . The specific form of this target matrix is further exploited to find the optimal boosting increment (i.e. the kernel evolution direction towards the next best major kernel alignment). Furthermore, it can be shown that the orthonormality of  $Q$  ensures the convergence of the learning algorithm.

## 2.3. One rank base kernels and weak functions

We consider base kernel functions defined by  $k_t(\mathbf{x}_i, \mathbf{x}_j) = f_t(\mathbf{x}_i)f_t(\mathbf{x}_j)$ , with  $f_t(\cdot)$  a function built by a weak learner. In order to build the combination, we work on a finite matrix on a given training set  $\mathbf{X}$ , which leads to the following expression::

$$\mathbf{K}_t = \sum_{s=1}^t \beta_s \mathbf{f}_s \mathbf{f}_s^\top = \mathbf{F}_t \mathbf{F}_t^\top,$$

with  $\mathbf{f}_t = f_t(\mathbf{X})$  and  $\mathbf{F}_t = \beta^{1/2} \odot (\mathbf{f}_1 \mathbf{f}_2 \dots \mathbf{f}_t)$  where  $\odot$  is the Hadamard product.

The values of weak functions  $f_t(\cdot)$  will be referred as supervised or semantic features built by the method.

We select base kernels iteratively in a boosting scheme:

$$\mathbf{K}_t = \mathbf{K}_{t-1} + \beta_t \mathbf{f}_t \mathbf{f}_t^\top \Leftrightarrow \mathbf{F}_t = (\mathbf{F}_{t-1} \beta_t^{1/2} \mathbf{f}_t),$$

where  $(\beta_t, f_t)$  is the solution of the following problem:

$$(\beta_t, f_t) = \arg \max_{\beta > 0, f} \mathcal{A}_{\mathbf{H}}(\mathbf{F}\mathbf{F}^\top + \beta \mathbf{f}\mathbf{f}^\top, \mathbf{Q}\mathbf{Q}^\top). \quad (1)$$

## 2.4. Weak targets

In order to train weak learners, we need to choose a target function  $f^*$ , which leads to the best alignment  $\mathcal{A}_{\mathbf{H}}(\mathbf{F}\mathbf{F}^\top, \mathbf{Q}\mathbf{Q}^\top)$ . In a two-classes context, it can be defined by  $f^*(\mathbf{x}_i) = 1$  if sample  $i$  is in the first class,  $-1$  otherwise. However, in the case of multi-class context, this is not obvious, since we need to spread each class data around equidistant centers [8].

We propose to consider the centers of (orthonormalized) classes in the space induced by the current combination kernel  $\mathbf{K}_t = \mathbf{F}_t \mathbf{F}_t^\top$ :

$$\mathbf{G}_t = \mathbf{Q}^\top \mathbf{F}_t.$$

The rows of  $\mathbf{G}_t$  are the coordinates of each class center.

The idea of our method is to move each center to make it equidistant from others. In [8] section 10.5, Vapnik states that the largest possible margin is achieved when the  $n_c$  vertices of  $(n_c - 1)$ -dimensional unitary simplex are centered onto the origin. A sufficient mean to achieve this properties is to build  $c$  orthonormal vertices, whose projection on a  $(n_c - 1)$  dimension space is the unitary simplex. In our case, that means that an ideal target set of class centers  $\mathbf{G}_t^*$  is such that  $\mathbf{G}_t^* (\mathbf{G}_t^*)^\top$  is proportional to the identity matrix  $\mathbf{Id}_{c,c}$ .

In next section we present an algorithm to compute these target functions.

## 3. ALGORITHM

### 3.1. Initialization

The first weak function target  $\mathbf{f}_0^*$  is defined as the first column of  $\mathbf{Q}$ . It is equivalent to set the first target centroids  $\mathbf{g}^*$  as one of the column of identity matrix. Then, we build a set of weak function candidates using least mean squares (LMS). The first weak function  $f_0$  is the candidate that maximizes the centered alignment  $\mathcal{A}_{\mathbf{H}}(\mathbf{f}\mathbf{f}^\top, \mathbf{Q}\mathbf{Q}^\top)$ .

### 3.2. Iterations

For each iteration  $t$ , we need to define a weak function target  $\mathbf{f}_t^*$ . The first  $t < n_c$  target function are defined as columns of matrix  $\mathbf{Q}$ ,  $\mathbf{f}_t^* = \mathbf{q}_t$ . Then target functions for  $t \geq n_c$  is defined as  $\mathbf{f}_t^* = \mathbf{Q}\mathbf{g}_t^*$ , with  $\mathbf{g}^*$  a vector based on the smallest eigenvalue  $\lambda$  and corresponding eigenvector  $\mathbf{v}$  of  $\mathbf{G}_t \mathbf{G}_t^\top$ :

$$\mathbf{g}^* = \sqrt{1 - \lambda} \sqrt{\frac{n_c}{\|\mathbf{H}\mathbf{F}_t \mathbf{F}_t^\top \mathbf{H}\|}} \mathbf{v}.$$

Then, we build a set of weak function candidates using least mean squares (LMS), and the weak function  $f_t$  is the one that maximize Eq. (1).

For a better generalization, we first evaluate the weak candidate function built using low dimension features. In the case

MKL	87.2±2.7
NLP- $\beta$	<b>87.9±1.8</b>
NLP- $\nu$ MC	87.8±2.1
NLP-B	87.3±2.7
MKL-prod	85.5 ± 1.2
MKL-avg ( $l_\infty$ )	84.9 ± 1.9
CF ( $l_\infty$ ) / AKM	86.7 ± 2.7
CG-Boost	84.8 ± 2.2
MKL (SILP or Simple) or OBSCURE	85.2 ± 1.5
LP- $\beta$	85.5 ± 3.0
LP-B	85.4 ± 2.4
MKL-FDA ( $l_p$ )	86.7 ± <b>1.2</b>
Proposed	<b>88.3 ± 1.1</b>

**Fig. 3:** Classification rate on Oxford Flower 17

where no one of them increases enough the centered alignment, we evaluate weak function candidates built using features with a larger dimension. This process is repeated until a relevant one is found.

### 3.3. Final supervised features

Final supervised features  $\mathbf{F}_T$  on the training set is then the concatenation of weighted weak functions values,  $\mathbf{F}_T = (\beta_t^{\frac{1}{2}} \mathbf{f}_t)_t$ .

Let us notice that we can compute these supervised feature for any image, as we learned functions  $f_t(\cdot)$  that can be evaluated outside the training set.

## 4. EXPERIMENTS

In this paper, we test our method on Oxford Flower17 and Flower102 databases [9]. In the following experiments, we use the same  $\chi^2$  distance matrices  $\mathbf{D}$  provided by the authors [9]. For each matrix, we compute a new matrix  $\mathbf{X} = e^{-\mu_f \mathbf{D}}$ , where  $\mu_f$  is set to be 1 over the mean value of the  $\chi^2$  distances over all the training images.

Then we reduce the dimension of  $\mathbf{X}$  by applying a PCA.

Our method then produce supervised features thanks to the training set and these supervised features are further used in a Gaussian Kernel SVM for each category.

Performances are measured with the same protocol as in [9].

### 4.1. Oxford Flower 17

Oxford Flower 17 is a database of 17 categories with 80 flower images per category. The dataset is made of 3 predefined random splits and each split consists in 3 sets: training (40 images per class), validation (20 images per class) and test (20 images per class). There are both large intra-category variance owing to appearance variations and small inter-category variance owing to partial appearance similarities between categories.

In our experiments, we use 7  $\chi^2$ -distance matrix provided by the authors of this base <sup>1</sup> to describe different properties of the flowers: colour, shape and texture histogram, HSV histogram, SIFT inside the foreground region, SIFT on the foreground region boundary and Histogram of Oriented Gradients.

We run our method with different settings. We build our classifier with features of different granularity. Initially, the coarsest features (i.e in our case the feature with low dimensions) are tried first. If the boosting algorithm fails to find a good kernel from these, then we consider more refined features with more information and detail. In practice, after PCA we keep features of 16, 32, 64, 128, 256, 512 and 1024 dimensions and we start our selection with low dimensions features. The results of this strategy are given in Table 2. This table gives the classification rate for several T and several max dimensions of feature. For example in case T=300 and  $\dim_{\text{feature}} = 128$ , we have selected up to 300 weak functions and the features have 16, 32, 64 or 128 dimensions. The method may stop by itself before reaching the desired number of iterations because no available feature increases enough the alignment. We can see that the average precision gets higher as the dimension of initial features and the number of iterations both increase.

We reported results from the literature [10] in Table 3. Our method outperforms these reference results, and gets lower standard deviation. This standard deviation is computed thanks to the 3 data split predefined by the authors of the dataset.

### 4.2. Oxford Flower 102

Oxford Flower 102 database contains 8189 images divided into 102 flower classes (cf. Fig.1). Each class consists in about 40 up to 250 images and the distribution of the images between classes is none uniform. The dataset is divided into three sets: training, validation and test. The training and validation sets consist in 10 images per class (1030 images) while the test set has 6129 images with at least 20 images per class.

Like [9] we use four different  $\chi^2$  distance matrices provided <sup>2</sup> to describe different properties of the flowers: HSV histogram, SIFT inside the foreground region, SIFT on the foreground region boundary and Histogram of Oriented Gradients.

In Fig.4, we compare our method with the methods from the literature and mainly to the reference method of the state-of-the-art [10]. As one can see, our method provides a significant gain.

<sup>1</sup><http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html>

<sup>2</sup><http://www.robots.ox.ac.uk/~vgg/data/flowers/102/index.html>



**Fig. 1:** Examples of categories for flower 102

	16	32	64	128	256	512	1024
50	$78.24 \pm 2.70$	$78.24 \pm 2.70$	$78.24 \pm 2.70$	$78.24 \pm 2.70$	$78.24 \pm 2.70$	$78.24 \pm 2.70$	$78.24 \pm 2.70$
100	$83.73 \pm 2.45$	$83.73 \pm 2.45$	$83.73 \pm 2.45$	$83.73 \pm 2.45$	$83.73 \pm 2.45$	$83.73 \pm 2.45$	$83.73 \pm 2.45$
200	$84.12 \pm 1.35$	$84.8 \pm 1.80$	$85.78 \pm 2.67$	$85.88 \pm 2.56$	$85.88 \pm 2.56$	$85.88 \pm 2.56$	$85.88 \pm 2.56$
300	-	$84.9 \pm 1.67$	$86.27 \pm 2.25$	$85.88 \pm 2.33$	$86.47 \pm 2.22$	$86.47 \pm 2.22$	$86.47 \pm 2.22$
400	-	-	$86.67 \pm 2.47$	$86.27 \pm 2.64$	$87.55 \pm 1.80$	$87.55 \pm 1.80$	$87.55 \pm 1.80$
500	-	-	-	$86.47 \pm 2.84$	$87.94 \pm 1.47$	<b><math>88.33 \pm 1.11</math></b>	<b><math>88.33 \pm 1.11</math></b>

**Fig. 2:** Classification rate on Flower 17 for several  $T$  and several dimensions of feature

MKL [9]	72.8
MKL [10]	73.4
NLP- $\beta$	<b>75.7</b>
NLP- $\nu$ MC	73.4
NLP-B	73.60
MKL-prod	73.8
MKL-avg	73.4
Proposed	<b>77.8</b>

**Fig. 4:** Classification rate on Oxford Flower 102

## 5. CONCLUSION

In this paper, we propose a novel algorithm to design multi-class kernel functions based on an iterative combination of weak kernels in a scheme inspired from boosting framework. The proposed method aims at building supervised features where the centroid for each class are optimally located. Experiments have been carried on state-of-the-art image databases Flower17 and Flower102. We compare our results with the reference methods, and showed that better results can be achieved using our method. We are currently working on a generalization of our method to collaborative learning context. Indeed, the same algorithm can target a kernel matrix for collaborative learning by considering that initial annotation matrix stores all previous retrieval runs.

## 6. REFERENCES

- [1] E. Chang, B. T. Li, G. Wu, and K.S. Goh, "Statistical learning for effective visual information retrieval," in *International Conference on Image Processing*, Barcelona, Spain, September 2003, pp. 609–612.
- [2] Y. Freund, R. Yyer, R.E. Schapire, and Y. Singer, "An efficient boosting algorithm for combining preferences," *Journal on Machine Learning Research*, vol. 4, pp. 933–969, November 2003.
- [3] M. I. Jordan F. Bach, G. R. G. Lanckriet, "Multiple kernel learning, conic duality, and the smo algorithm," in *International Conference on Machine Learning*, 2004.
- [4] P. Gehler and S. Nowozin, "On feature combination for multiclass object classification," in *Computer Vision, 2009 IEEE 12th International Conference on*, 29 2009-oct. 2 2009, pp. 221–228.
- [5] F. Orabona and L. Jie, "Ultra-fast optimization algorithm for sparse multi kernel learning," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, Lise Getoor and Tobias Scheffer, Eds., New York, NY, USA, June 2011, ICML '11, pp. 249–256, ACM.
- [6] K. Crammer, J. Keshet, and Y. Singer, "Kernel design using boosting," in *Advances in Neural Information Processing Systems*. 2003, pp. 537–544, MIT Press.
- [7] M. Kawanabe, S. Nakajima, and A. Binder, "A procedure of adaptive kernel combination with kernel-target alignment for object classification," in *ACM International Conference on Image and Video Retrieval*, 2009.
- [8] V. Vapnik, *Estimation of Dependences Based on Empirical Data*, Springer-Verlag, 1982.
- [9] M-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.
- [10] K. Mikolajczyk M. Awais, F. Yan and J. Kittler, "Augmented kernel matrix vs classifier fusion for object recognition," in *Proceedings of the British Machine Vision Conference*. 2011, pp. 60.1–60.11, BMVA Press.